Our File No. 9281-4144
Client Reference No. J US98135

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
APPLICATION FOR UNITED STATES LETTERS PATENT

INVENTOR: Takayuki Sugawara

TITLE: Arithmetic Operation Unit Suitable for Correcting Lost Data by General-Purpose Computer

ATTORNEY: Gustavo Siller, Jr.
BRINKS HOFER GILSON & LIONE
P.O. BOX 10395
CHICAGO, ILLINOIS 60610
(312) 321-4200

EXPRESS MAIL NO. EL 746 760 130 US

DATE OF MAILING 8/8/01

ARITHMETIC OPERATION UNIT SUITABLE FOR CORRECTING LOST DATA BY GENERAL-PURPOSE COMPUTER

BACKGROUND OF THE INVENTION

1. Field of the Invention

The present invention relates to an arithmetic operation unit capable of multiplying elements of a Galois field at a high speed, and more specifically, to an arithmetic operation unit suitable for in a general-purpose computer for correcting lost data.

2. Description of the Related Art

In error correction processing that is executed when data is recorded in, and reproduced from, a recording medium, check symbols E_0, E_1, E_2, ... are encoded in the recording medium as additional codes. When the data is reproduced, symbols S_0, S_1, S_2, ... are determined by ExOring data (user symbols) with the check symbols E_0, E_1, E_2, ... Then, the magnitude of any error is calculated by subjecting the symbols to an arithmetic operation.

While these symbols E_0 and S_0 can be determined by the calculation of simple exclusive OR, power calculation of α is necessary for symbols E_1 and S_1 or higher. Here, α is defined as an element when a primitive polynomial G(X) on a Galois field is made to 0.

Since a recording-reproducing apparatus and the like is not provided with an arithmetic operating section dedicated for a Galois field, it determines the check symbol E and the symbol

S in decoding by means of a look-up table.

Hitherto, a table of 8 bits (256) × 8 bits (256) = 64k bytes is used as the look-up table. Thus, it is possible to determine the check symbol E and the symbol S in decoding as to data each having 8 bits. However, since computers presently have an access width of a unit of 16 bits or 32 bits, it is preferable to execute the arithmetic operation of the symbols by the unit of 16 bits or 32 units.

However, the arithmetic operation of the unit of 16 bits, for example, requires a table of 16 bits (64k bytes) × 16 bits (64k bytes) = 4G bytes, the construction of which is a practical impossibility.

Further, while to the use a power expression conversion table has also been contemplated, this similarly requires a table on the order of several hundreds of kilobytes for a unit of 16 bits or larger.

SUMMARY OF THE INVENTION

An object of the present invention, which has been made to solve the above conventional problem, is to provide an arithmetic operation unit capable of executing power calculation using the element α of an Galois field at a high speed and correcting, for example, lost data at a high speed.

The present invention is characterized in an arithmetic operation unit for multiplying α^i when the element of $G(X) = 0$ of a primitive polynomial $G(X)$ on a Galois field represented by the following Expression (1) is represented by α, the arithmetic

operation unit comprising a shift operating section for shifting elements by i bits before multiplication and a referring section for referring to a look-up table of 2^i pieces of elements when the multiplier of α is represented by i.

$$G_{(x)} = g_m x^m + g_{m-1}x^{m-2} + g_{m-2}x^{m-2} + \cdots + g_{p+1}x^{p+1} + g_p x^p + \cdots + g_0 \quad \cdots (1)$$

Further, the present invention relates to an arithmetic operation unit for calculating $U \cdot \alpha^i$ based on an element U on the Galois field represented by the following Expression (2) when the element of $G_{(X)} = 0$ of a primitive polynomial $G_{(X)}$ on a Galois field represented by the above Expression (1) is represented by α, wherein a shift operating section for shifting the element of the U by i bits is ExOred with a referring section for referring to a look-up table having 2^i pieces of elements according to the least significant i bits of the U.

$$U = \alpha^n u_n + \alpha^{n-1}u_{n-1} + \cdots + \alpha^2 u_2 + \alpha^1 u_1 + u_0 \quad \cdots (2)$$

Further, when data (user symbols) is represented by D_1, D_2, ..., D_k, error check symbols represented by the following Expression (3) are calculated.

$$
\begin{aligned}
&D_1+D_2 + D_3 + \cdots + D_{k-1} + D_k = E_0 \\
&\alpha^k D_1 + \alpha^{k-1} D_2 + \alpha^{k-2} D_3 + \cdots + \alpha^2 D_{k-1} + \alpha D_k = E_1 \\
&\alpha^{(k)^2} D_1 + \alpha^{(k-1)^2} D_2 + \alpha^{(k-2)^2} D_3 + \cdots + \alpha^4 D_{k-1} + \alpha^2 D_k = E_2 \quad \cdots (3) \\
&\qquad\vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\alpha^{(k)^{n-k-1}} D_1 + \alpha^{(k-1)^{n-k-1}} D_2 + \cdots + \alpha^{n-k} D_{k-1} + \alpha^{n-k-1} D_k = E_{n-k-1}
\end{aligned}
$$

In addition to the above, when data (user symbols) is decoded, symbols S_0, S_1, S_2, …, S_{n-k-1} are obtained by calculating the following Expression (4).

$$
\begin{aligned}
&D_1+D_2 + D_3 + \cdots + D_{k-1} + D_k + E_0 = S_0 \\
&\alpha^k D_1 + \alpha^{k-1} D_2 + \alpha^{k-2} D_3 + \cdots + \alpha^2 D_{k-1} + \alpha D_k + E_1 = S_1 \\
&\alpha^{(k)^2} D_1 + \alpha^{(k-1)^2} D_2 + \alpha^{(k-2)^2} D_3 + \cdots + \alpha^4 D_{k-1} + \alpha^2 D_k + E_2 = S_2 \\
&\qquad\vdots \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \vdots \\
&\alpha^{(k)^{n-k-1}} D_1 + \alpha^{(k-1)^{n-k-1}} D_2 + \cdots + \alpha^{n-k} D_{k-1} + \alpha^{n-k-1} D_k + E_{n-k-1} = S_{n-k-1} \\
&\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad \cdots (4)
\end{aligned}
$$

Further, when the magnitude of an error is determined using the symbols S_0, S_1, S_2, …, S_{n-k-1}, the magnitude of the error is determined by providing the following inverse element reference table and referring to the table.

a) α^1, α^2, … α^k

b) $1 + \alpha^1$, $1 + \alpha^2$, … $1 + \alpha^k$

BRIEF DESCRIPTION OF THE DRAWINGS

FIG. 1 is a block diagram showing an example of an α multiplication circuit.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

The present invention provides an arithmetic operation unit capable of multiplying α^i at a high speed when the element of a primitive polynomial G(X) on a Galois field, which has a dimension in coincidence with the access width (for example, 16 bits, 32 bits, and the like) of a computer, is represented by α. The arithmetic operation unit of the present invention can execute multiplication even if the computer has a large access width. Thus, the arithmetic operation unit of the present invention correct an error in a unit of larger than 8 bits, thereby overcoming the limitations of a conventional arithmetic operation unit.

Table 1 shows a format, which is represented in a unit of one block, of data recorded in a magnetic disc or the like.

Table 1

Packet No	Double Word 0	Double Word 1	Double Word 2		Double Word 127
0	Byte 0-3(D_1)	Byte 4-7	Byte 8-11	.....	Byte 508-511
1	Byte 0-3(D_2)	Byte 4-7	Byte 8-11	.....	Byte 508-511
.....	.....	.....	.....	.....	.....
63	Byte 0-3(D_{63})	Byte 4-7	Byte 8-11	.....	Byte 508-511
64	ECC[0]	ECC[0]	ECC[0]	.....	ECC[0]
65	ECC[1]	ECC[1]	ECC[1]	.....	ECC[1]

Table 1 shows Packet Nos. in a longitudinal direction, wherein Nos. 0 to 63 show user blocks including data (user symbols). Packet Nos. 64 and 65 include check symbols E_0 and E_1 for correcting errors. These check symbols E_0 and E_1 are codes for correcting errors.

The data from Packet Nos. 0 to 63 is arranged in a double word unit having 32 bits (4 bytes). An arithmetic operation is executed from Packet Nos. 0 to 63 so as to calculate the check symbol E_0 in each double word unit, and a result of the calculation is stored in Packet No. 64. Further, the check symbol E_1 is calculated in each double word unit, and a result of the calculation is stored in Packet No. 65.

When data is recorded in a recording medium through the format shown by Table 1, the check symbol E_0 is calculated by each double word unit. A general expression of the calculation is shown by the following Expression (5). In the following Expression (5), D_1, D_2, D_3, …, D_k denote user symbols of Packet Nos. 0 to 63 in each double word unit, and each user symbol is composed of, for example, 32 bits. Note that in the case of Table 1, the number k of the above symbols is 63. E_0, E_1, E_2, …, E_{n-k-1} denote the check symbols, and the number of the check symbols is n - k. It should be noted that Table 1 shows a case in which number of the check symbols is 2.

$$
\begin{aligned}
&D_1+D_2+D_3+\cdots+D_{k-1}+D_k=E_0\\
&\alpha^k D_1+\alpha^{k-1}D_2+\alpha^{k-2}D_3+\cdots+\alpha^2 D_{k-1}+\alpha D_k=E_1\\
&\alpha^{(k)^2}D_1+\alpha^{(k-1)^2}D_2+\alpha^{(k-2)^2}D_3+\cdots+\alpha^4 D_{k-1}+\alpha^2 D_k=E_2 \qquad \cdots(5)\\
&\qquad\vdots \qquad\qquad\qquad\qquad\qquad\qquad\vdots\\
&\alpha^{(k)^{n-k-1}}D_1+\alpha^{(k-1)^{n-k-1}}D_2+\cdots+\alpha^{n-k}D_{k-1}+\alpha^{n-k-1}D_k=E_{n-k-1}
\end{aligned}
$$

Next, when the data is decoded from the recording medium, the arithmetic operation of the following Expression (6) is executed. As shown below, the magnitude of a data error is determined from symbols S_0, S_1, S_2, …, S_{n-k-1} which are determined by this arithmetic operation.

$$
\begin{aligned}
&D_1+D_2+D_3+\cdots+D_{k-1}+D_k+E_0=S_0\\
&\alpha^k D_1+\alpha^{k-1}D_2+\alpha^{k-2}D_3+\cdots+\alpha^2 D_{k-1}+\alpha D_k+E_1=S_1\\
&\alpha^{(k)^2}D_1+\alpha^{(k-1)^2}D_2+\alpha^{(k-2)^2}D_3+\cdots+\alpha^4 D_{k-1}+\alpha^2 D_k+E_2=S_3\\
&\qquad\vdots \qquad\qquad\qquad\qquad\qquad\qquad\vdots\\
&\alpha^{(k)^{n-k-1}}D_1+\alpha^{(k-1)^{n-k-1}}D_2+\cdots+\alpha^{n-k}D_{k-1}+\alpha^{n-k-1}D_k+E_{n-k-1}=S_{n-k-1}
\end{aligned}
\qquad \cdots(6)
$$

In encoding and decoding, the arithmetic operations of Expressions (5) and (6) can be executed with an α^i multiplier having a symbol number up to -1. Note that i shows numbers 0, 1, 2, 3, …, k. It should be noted that the symbol "+" in the above Expression (1) or later expressions shows an exclusive OR (ExOR).

First, while the check symbols E_0 and S_0 in Expressions (5) and (6) can simply be calculated by exclusive OR, power calculation is necessary to calculate E_1, S_1 and later. FIG. 1 shows an example of an arithmetic operation circuit for executing

this calculation. As shown in FIG. 1, D_1, D_2, D_3, …, D_k are stored in a register, and each of them is subjected to power calculation of α, α^2, …, α^{n-k-1} to thereby calculate E_1, E_2, …, E_{n-k-1} and S_1, S_2, …, S_{n-k-1}.

The calculation method and the arithmetic operation unit of the present invention for increasing the speed of the power calculation will now be described.

The primitive polynomial GF(2) of a Galois field is represented by the following Expression (7).

$$G_{(x)} = g_m x^m + g_{m-1}x^{m-2} + g_{m-2}x^{m-2} + \cdots + g_{p+1}x^{p+1} + g_p x^p + \cdots + g_0 \quad \ldots(7)$$

The primitive polynomial GF(2) of the Galois field will be described using the following Expression (8) as an example so that the principle of the arithmetic operation unit can be explained in simplified terms.

$$G_{(x)} = X^8 + X^4 + X^3 + X^2 + 1 \quad \ldots(8)$$

In Expressions (4) and (5), when symbols corresponding to the user symbols D_1, D_2, D_3, …, D_k are represented by u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0, each composed of 8 bits, U (u_7, u_6, u_5, u_4, u_3, u_2, u_1, u_0) is represented by the following Expression (9).

$$U = \alpha^7 u_7 + \alpha^6 u_6 + \alpha^5 u_5 + \alpha^4 u_4 + \alpha^3 u_3 + \alpha^2 u_2 + \alpha^1 u_1 + u_0 \quad \ldots(9)$$

Further, $\alpha \cdot U$ is represented by the following Expression (10).

$$\alpha \cdot U = \alpha^8 u_7 + \alpha^7 u_6 + \alpha^6 u_5 + \alpha^5 u_4 + \alpha^4 u_3 + \alpha^3 u_2 + \alpha^2 u_1 + \alpha u_0 \quad \ldots (10)$$

The $\alpha \cdot U$ represented by Expression (10) is equal to what is obtained by shifting u_7, u_6, u_5, u_4, u_3, u_2, u_1 and u_0 by 1 bit as to the elements of Expression (9) so as to obtain the following Expression (11), and by adding (ExORing) Expression (12) obtained from the primitive polynomial of Expression (8) to Expression (11).

$$\alpha^7 u_6 + \alpha^6 u_7 + \alpha^5 u_4 + \alpha^4 u_3 + \alpha^2 u_1 + \alpha^1 u_0 \quad \ldots (11)$$

$$\alpha^8 (= \alpha^4 + \alpha^3 + \alpha^2 + 1) \cdot u_7 \quad \ldots (12)$$

Accordingly, the arithmetic operation from U to $\alpha \cdot U$ can be executed at a high speed by preparing two types of look-up tables (reference sections, Table 2) for a case in which the least significant bit u_7 of U is "1" and a case in which it is "0", together with a shift operating section.

Table 2

u7	Value of look-up table
0	0
1	$\alpha^4+\alpha^3+\alpha^2+\alpha^1$

Next, $\alpha^2 \cdot U$ is represented as shown by the following

Expression (13).

$$\alpha^2 \cdot U = \alpha^9 u_7 + \alpha^8 u_6 + \alpha^7 u_5 + \alpha^6 u_4 + \alpha^5 u_3 + \alpha^4 u_2 + \alpha^3 u_1 + \alpha^2 u_0 \quad \ldots(13)$$

This is equal to what is obtained by shifting the elements of U of Expression (9) by 2 bits and by adding (ExORing) the following Expression (14) to the resultant expression.

$$\alpha^9 (= \alpha \cdot (\alpha^4 + \alpha^3 + \alpha^2 + 1)) \cdot u_7 + \alpha^8 (= \alpha^4 + \alpha^3 + \alpha^2 + 1) \cdot u_6 \quad \ldots(14)$$

That is, it is sufficient to prepare 4 ($= 2^2$) types of look-up tables (Table 3) according to the least significant 2-bit values of the elements of U, to shift U by 2 bits, and to subject the resultant expression to ExORing.

Table 3

u6,u7	Value of look-up table
00	0
01	$\alpha^4+\alpha^3+\alpha^2+\alpha^1$
10	$\alpha(\alpha^4+\alpha^3+\alpha^2+\alpha^1)$
11	$\alpha(\alpha^4+\alpha^3+\alpha^2+\alpha^1)$ $+\alpha^4+\alpha^3+\alpha^2+\alpha^1$

Since the shift operation is generally included in CPO and the look-up tables can be realized by a memory of a small capacity, the shift operation is a generally applicable and high speed

arithmetic operation method.

The aforementioned procedure can be employed also in $\alpha^3 \cdot U$, $\alpha^4 \cdot U$ and so on in the same manner. That is, a high speed calculation of α^i can be executed by providing: 1) an i-bit shift section; and 2) 2^i types of look-up tables (referring sections).

Next, an error correction executed by the symbols S_0, S_1, S_2, ..., S_{n-k-1} will be described.

When the data reproduced from the recording medium has no error, all of the S_0, S_1, S_2, ..., S_{n-k-1} in Expression (6) are 0. Further, when an error arises in the user symbols, the magnitude of the error can be calculated by the following arithmetic operation if lost data, in which a portion where an error arises is previously known, is to be corrected. For example, when it is assumed that errors in the user symbols arise at i-th and j-th positions from a rear side and that the magnitudes of the errors are represented by e_i and e_j, the relationship between the magnitudes of the errors and the symbols S_0 and S_1 is represented by the following Expression (15).

$$\begin{aligned} e_i + e_j &= S_0 \\ \alpha^i e_i + \alpha^j e_j &= S_1 \end{aligned} \qquad \ldots(15)$$

When e_i and e_j are determined from Expression (15), they are represented by Expression (16), which can be solved as simultaneous equations with two unknowns.

$$e_i = \frac{\alpha^j \cdot S_0 \cdot S_1}{\alpha^i + \alpha^j}$$
$$e_j = \frac{(\alpha^j \cdot S_0 \cdot S_1) \cdot \alpha^{-i}}{1 + \alpha^{j-1}} \qquad \ldots(16)$$

Next, since the occurrence of an error e_j in the check symbol E_0 results in the following expression 17, the magnitude of the error e_i can be solved by a linear equation.

$$e_i + e_j = S_0$$
$$\alpha^i e_i = S_1 \qquad \ldots(17)$$

Similarly, when errors e_i, e_j, and e_k arise at three position of the user symbols, the magnitude of the error e_i (Expression (19)) can be determined by solving simultaneous equations with three unknowns shown in Expression (18).

$$e_i + e_j + e_k = S_0$$
$$\alpha^i e_i + \alpha^j e_j = S_1 \qquad \ldots(18)$$
$$\alpha^{2i} e_i + \alpha^{2j} e_j = S_2$$

$$e_i = \frac{\alpha^j \cdot \alpha^k \cdot S_0 \cdot \alpha^j \cdot S_1 \cdot \alpha^k \cdot S_1 + S_2}{(-\alpha^i + \alpha^j)(-\alpha^i + \alpha^k)}$$
$$e_j = \frac{(\alpha^j \alpha^k S_0 \alpha^j S_1 \alpha^k S_1 + S_2)\alpha^{-j-k}}{(1 - \alpha^{i-j})(1 - a^{i-k})} \qquad \ldots(19)$$

e_j and e_k can be calculated in the same way.

In the above arithmetic operation, as the degree of the primitive polynomial increases, an amount of inverse elements

to be calculated is increased. Thus, it is advantageous to deform the primitive polynomial so that it is within a predetermined value and to refer to inverse elements by a look-up table system from a view point of speed. As to the inverse elements, it is sufficient to provide the following two types of tables (referring section):

a) α^1, α^2, ..., α^k (k: number of user symbols), and

b) $1 + \alpha^1$, $1 + \alpha^2$, ..., $1 + \alpha^k$ (k: number of user symbols).

As described above, according to the present invention, it is possible to execute an arithmetic operation for error correction using the primitive polynomial of a Galois field.